Expert Systems with Applications 37 (2010) 5682–5687

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Chaotic bee colony algorithms for global numerical optimization

Bilal Alatas [*]

Firat University, Faculty of Engineering, Department of Computer Engineering, 23119 Elazig, Turkey

A R T I C L E   I N F O

Keywords:
Bee colony algorithm
Global numerical optimization
Chaos

A B S T R A C T

Artificial bee colony (ABC) is the one of the newest nature inspired heuristics for optimization problem. Like the chaos in real bee colony behavior, this paper proposes new ABC algorithms that use chaotic maps for parameter adaptation in order to improve the convergence characteristics and to prevent the ABC to get stuck on local solutions. This has been done by using of chaotic number generators each time a random number is needed by the classical ABC algorithm. Seven new chaotic ABC algorithms have been proposed and different chaotic maps have been analyzed in the benchmark functions. It has been detected that coupling emergent results in different areas, like those of ABC and complex dynamics, can improve the quality of results in some optimization problems. It has been also shown that, the proposed methods have somewhat increased the solution quality, that is in some cases they improved the global searching capability by escaping the local solutions.

1. Introduction

An optimization problem is modeled in such a way that a classical algorithm, which requires several assumptions and/or modifications which might not be easy to validate in many situations, can handle it. These assumptions and/or modifications on the original optimization problem parameters (rounding variables, softening constraints, etc.) certainly affect the solution quality (Baykasoglu, Ozbakir, & Tapkan, 2007). They are insufficient if integer and/or discrete decision variables are required in optimization models (Baykasoglu et al., 2007). Namely, classical optimization algorithms are inflexible to adapt the solution procedure to an optimization problem.

Furthermore, solution strategies of classical optimization algorithms are generally depended on the type of objective and constraint functions (linear, non-linear, etc.) and the type of variables used in the problem modeling (integer, real, etc.). Their efficiency is also very much dependent on the size of the solution space, number of variables and constraints used in the problem modeling, and the structure of the solution space (convex, non-convex, etc.). Namely, they do not offer general solution strategies that can be applied to problem formulations where, different type of variables, objective and constraint functions are used. However, most of the optimization problems require different types of variables, objective and constraint functions simultaneously in their formulation (Baykasoglu et al., 2007).

Inefficiency of classical optimization algorithms in solving larger scale and/or highly non-linear problems forced researchers to find more flexible and adaptable, problem and model independent, general purpose heuristic algorithms. These algorithms are efficient and flexible and they can be modified and/or adapted to suit specific problem requirements. Researches on these algorithms are still continuing all around the globe. Fig. 1 shows the classifications of the heuristic algorithms.

Swarm intelligence that combines of biology and social based heuristics has become a research interest to many research scientists of related fields in recent years (Abbass, 2001). Particle swarm optimization, ant colony optimization, and bee colony algorithms can be considered as subfields of swarm intelligence. A few models have been developed to model the intelligent behaviors of honeybee swarms and applied for solving the problems (Abbass, 2001; Karaboga, 2005; Karaboga & Basturk, 2008; Pai, Yang, & Chang, 2009; Pham et al., 2006; Yang, 2005). Recently proposed artificial bee colony (ABC) algorithm has been inspired by the intelligent behavior of real honey bees and proven to be a better heuristic for global numerical optimization (Karaboga, 2005; Karaboga & Basturk, 2008).

Many chaotic maps in the literature possess certainty, ergodicity and the stochastic property. Recently, chaotic sequences have been adopted instead of random sequences and very interesting and somewhat good results have been shown in many applications. They have also been used together with some heuristic optimization algorithms (Alatas, Akin, & Ozer, 2009; Coelho & Mariani, 2008) to express optimization variables. The choice of chaotic sequences is justified theoretically by their unpredictability, i.e., by their spread-spectrum characteristic, non periodic, complex temporal behavior, and ergodic properties.

In this paper, sequences generated from different chaotic systems substitute random numbers for different parameters of ABC

* Tel.: +90 424 237 00 00/6307.
E-mail address: balatas@firat.edu.tr

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.02.042

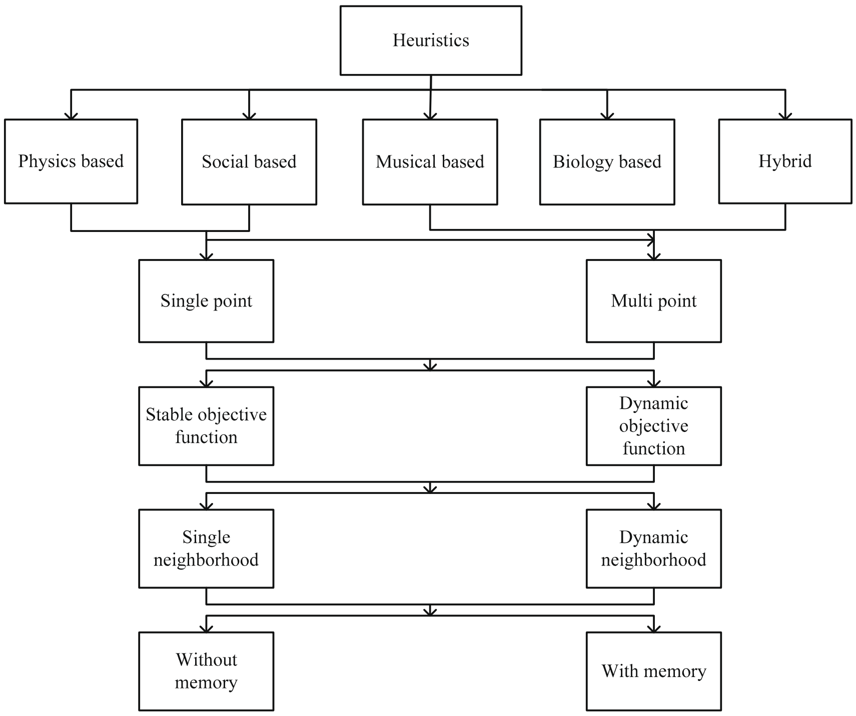

Fig. 1. Heuristic algorithms.

where it is necessary to make a random-based choice. For this purpose, different ABC methods that use chaotic maps as efficient alternatives to pseudorandom sequences have been proposed. By this way, it is intended to enhance the global convergence and to prevent to stick on a local solution. However, in general, it is hard to estimate how good most chaotic random number generator by applying statistical tests are, as they do not follow the uniform distribution. The simulation results show that the application of deterministic chaotic signals instead of random sequences may be a possible strategy to improve the performances of ABC.

The remaining of this paper is organized as follows. Review of ABC is summarized in Section 2. Section 3 describes the proposed methods, Chaotic bee colony algorithms, shortly CBCAs. Section 4 describes the benchmark problems used for comparisons of the proposed methods. In Section 5, the testing of the proposed methods through benchmark problems are carried out and the simulation results are compared. Finally, the conclusions are drawn based on the comparison analysis reported and presented in Section 6.

2. Artificial bee colony algorithm

ABC algorithm has been inspired by the intelligent behavior of real honey bees (Karaboga, 2005; Karaboga & Basturk, 2008). In the algorithm, the artificial bee colony consists of three groups of bees: employed bees, onlookers and scouts. The first half of the colony consists of the employed artificial bees and the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source has been exhausted by the bees becomes a scout. The pseudo-code of the algorithm is given in Fig. 2.

Each cycle of the search consists of three steps after initialization step: moving the employed bees onto the food sources and calculating their nectar amounts; placing the onlookers onto the food sources and calculating the nectar amounts; and determining the scout bees and directing them onto possible food sources (Karaboga, 2005; Karaboga & Basturk, 2008). A food source position represents a possible solution of the problem to be optimized. The amount of nectar of a food source corresponds to the quality of the solution represented by that food source. Each employed bee is moved onto her food source area for determining a new food source within the neighborhood of the present one, and then its nectar amount is evaluated. If the nectar amount of the new one is higher, then bee forgets the previous and memorizes the new one. Onlookers are placed on the food sources by using a probability based selection process. As the nectar amount of a food source increases, the probability value with which the food source is preferred by onlookers increases similar to the natural selection process in evolutionary algorithms (Karaboga, 2005; Karaboga & Basturk, 2008).

Every bee colony has scouts considered as the colony's explorers that do not have any guidance while looking for food. They are primarily concerned with finding any kind of food source. As a result of such behavior, the scouts are characterized by low search costs and a low average in food source quality. Occasionally, the scouts can accidentally discover rich, entirely unknown food sources. In the case of artificial bees, the artificial scouts could have the fast discovery of the group of feasible solutions as a task. In ABC, one of the employed bees is selected and classified as the scout bee (Karaboga, 2005; Karaboga & Basturk, 2008). The selection is controlled by a control parameter called *limit*. If a solution representing a food source is not improved by a predetermined number of trials, then that food source is abandoned by its employed bee and the employed bee of this food becomes a scout. The number of trials for releasing a food source is equal to the value of *limit* which is an important control parameter of ABC. In a robust search process exploration and exploitation processes must be carried out together. In the ABC algorithm, while onlookers and employed bees carry out the exploitation process in the search space, the scouts control the exploration process. These three steps are repeated until the termination criteria are satisfied (Karaboga, 2005; Karaboga & Basturk, 2008).

```
-Initialize problem parameters
-Initialize algorithm parameters
-Construct initial Employed Bee Colony solutions
-Evaluate fitness value for each bee
-$I$=0
-Repeat
   -$N$=0
   -Repeat
      $k$ = a solution in the neighbourhood of $i$
      $\Phi$ = a random number in the range [-1,1]
      -Produce new solutions (food source positions) $\upsilon_{i,j}$ in the
      neighbourhood of $x_{i,j}$ for the employed bees using the formula
      $\upsilon_{i,j} = x_{i,j} + \Phi_{ij}(x_{i,j} - x_{k,j})$
      -Apply the greedy selection process between $x_i$ and $\upsilon_i$
      -Calculate the probability values $P_i$ for the solutions $x_i$ by means of
      their fitness values using the equation
      $P_i = \frac{fit(i)}{\sum fit(i)}$
      -Assign Onlooker Bees (produce new solutions(positions) $\upsilon_i$ to Employed
      Bees ($x_i$) according to probabilities and evaluate them
      For all Onlooker Bees Do
         -Apply the greedy selection process for the onlookers between $x_i$ and
         $\upsilon i$
         If fit(Best Onlooker)<fit(Employed)
            -Replace employed bee solution with respective onlooker solution
         End If
      End Do
      If fit(BestFeasible Onlooker)<fit(Best)
         -Find best Feasible Onlooker, replace with Best solution
      End If
      $N$=$N$+1
   Until ($N$=Number of Employed Bee)
   -Determine the abandoned solution (source), if exists, and replace it
   with a new randomly produced solution $x_i$ for the scout using the equation
      $x_{i,j}$=min$_j$+rand(0,1)*(max$_j$-min$_j$)
   -The worst employed bees as many as the number of scout bees in the
   population are respectively compared with the scout solutions. If the
   scout solution is better than employed solution, employed solution is
   replaced with scout solution. Else employed solution is transferred to
   the next cycle without any change
   -$I$=$I$+1
Until ($I$=MaxIteration)
```

Fig. 2. Pseudo-code of ABC algoritm.

The flowchart of the ABC algorithm is given in Fig. 3

3. Chaotic bee colony algorithms

In simulating complex phenomena, sampling, numerical analysis, decision making and especially heuristic optimization needs random sequences with a long period and good uniformity (Coelho & Mariani, 2008; Schuster, 1988). Chaos is a deterministic, random-like process found in non-linear, dynamical system, which is non-period, non-converging and bounded. Moreover, it has a very sensitive dependence upon its initial condition and parameter (Coelho & Mariani, 2008; Schuster, 1988). The nature of chaos is apparently random and unpredictable and it also possesses an element of regularity. Mathematically, chaos is randomness of a simple deterministic dynamical system and chaotic system may be considered as sources of randomness (Coelho & Mariani, 2008; Schuster, 1988).

A chaotic map is a discrete-time dynamical system

$$x_{k+1} = f(x_k), \quad 0 < x_k < 1, \quad k = 0, 1, 2, \ldots \tag{1}$$

running in chaotic state. The chaotic sequence

$$\{x_k : k = 0, 1, 2, \ldots\}$$

can be used as spread-spectrum sequence as random number sequence.

Chaotic sequences have been proven easy and fast to generate and store, there is no need for storage of long sequences (Heidari-Bateni & McGillem, 1994). Merely a few functions (chaotic maps) and few parameters (initial conditions) are needed even for very long sequences. In addition, an enormous number of different sequences can be generated simply by changing its initial condition. Moreover these sequences are deterministic and reproducible.

Recently, chaotic sequences have been adopted instead of random sequences and very interesting and somewhat good results have been shown in many applications such as secure transmission (Suneel, 2006; Wong, Man, Li, & Liao, 2005), and nonlinear circuits (Arena, Caponetto, Fortuna, Rizzo, & La Rosa, 2000), DNA computing (Manganaro & de Gyvez, 1997), image processing (Han, Hu, Yu, & Wang, 2007). The choice of chaotic sequences is justified

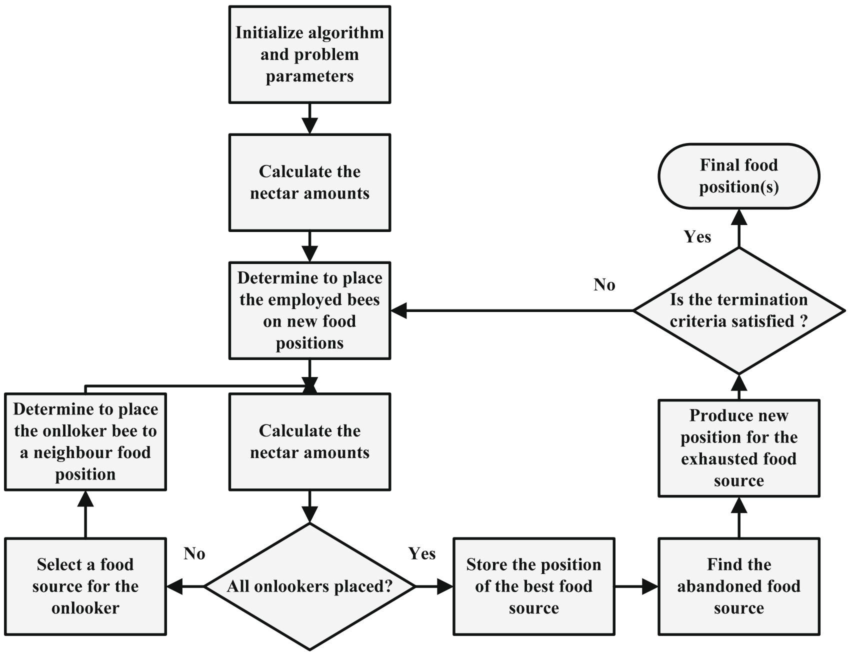
Initialize algorithm and problem parameters
Calculate the nectar amounts
Determine to place the employed bees on new food positions
Determine to place the onlloker bee to a neighbour food position
Calculate the nectar amounts
Select a food source for the onlooker
No
All onlookers placed?
Yes
Store the position of the best food source
Find the abandoned food source
Produce new position for the exhausted food source
Is the termination criteria satisfied ?
No
Yes
Final food position(s)

Fig. 3. The flowchart of the ABC algorithm.

theoretically by their unpredictability, i.e., by their spread-spectrum characteristic and ergodic properties.

Randomly initializing of ABC and the *limit* parameter that is adjusted in initialization step that cannot be changed during new iterations may affect the algorithm performance on convergence speed. This paper provides new approaches introducing chaotic maps with ergodicity, irregularity and the stochastic property in ABC to improve the global convergence by escaping the local solutions. The use of chaotic sequences in ABC can be helpful to escape more easily from local minima than can be done through the classical ABC. The selected chaotic maps tat produce chaotic numbers in (0, 1) for the experiments have been listed in Table 1.

New chaotic ABC (CABC) algorithms may be simply classified and described as follows:

3.1. Chaotic ABC 1 (CABC1)

Initial artificial colony is generated by iterating the selected chaotic maps until reaching to the colony size as shown in Fig. 4. N is the dimension for the problem; i is the colony member; and j is the dimension. $x_{i,j}$ is the jth dimension of the ith colony member

3.2. Chaotic ABC 2 (CABC2)

In this algorithm, if a solution representing a food source is not improved by *limit*/2 trials, then that food source is abandoned by its employed bee and the scout of this employed bee starts chaotic search for *limit*/2 iterations. $x_{i,j}$ is the jth dimension of the ith colony member. $c_{i,j}$ is the chaotic number generated by treating the

Table 1
Definitions of the used chaotic maps.

Name	Definition
Logistic map	$X_{n+1} = 4X_n(1 - X_n)$
Circle map	$X_{n+1} = X_n + 1.2 - (0.5/2\pi)\sin(2\pi X_n)\mathrm{mod}(1)$
Gauss map	$X_{n+1} = \begin{cases} 0, & X_n = 0 \\ 1/X_n\mathrm{mod}(1), & X_n \in (0,1), \end{cases} 1/X_n\mathrm{mod}(1) = \frac{1}{X_n} - \left\lfloor \frac{1}{X_n} \right\rfloor$
Henon map	$X_{n+1} = 1 - 1.4X_n^2 + 0.3X_{n-1}$
Sinusoidal iterator	$X_{n+1} = \sin(\pi x_n)$
Sinus map	$X_{n+1} = 2.3(X_n)^{2Sin(\pi X_n)}$
Tent map	$X_{n+1} = \begin{cases} X_n/0.7, & X_n < 0.7 \\ 10/3X_n(1 - X_n), & \text{otherwise} \end{cases}$

```
CI = the maximum number of chaotic iteration
i = 0
Repeat
    Randomly initialize the first chaotic variable
    j = 0
    Repeat
        Generate chaotic variable $cm_{i,j}$ according to the selected map
        $x_{i,j} = x_j^{\min} + cm_{i,j} \times (x_j^{\max} - x_j^{\min})$
        j = j + 1
    Until (j < N)
    i = i + 1
Until (i ≤ ColonySize)
```

Fig. 4. Pseudo-code of CABC1 change to original ABC.

```
Initialize the first chaotic variable for search
t = 0
Repeat
    Generate chaotic variable $cm_{i,j}$ iterating the selected map
    Map $cm_{i,j}$ back to the range around original value with radius r using
    $x_{i,j} = x_{i,j} + \frac{(x_{i,j}^{\max} - x_{i,j}^{\min})}{2} \times (2 \times cm_{i,j} - 1)$
    Evaluate the fitness of new $x_i$
    If better solution is found
        Replace the relevant dimension of better solution
        Switch to another dimension
    End If
    t = t + 1
Until (t > limit / 2)
```

Fig. 5. Pseudo-code of chaotic search for the bees.

Table 2
Properties of test problems, *lb* indicates lower bound, *ub* indicates upper bound, *opt* indicates optimum point.

Function no	Function name	Definition	*lb*	*ub*	*opt*	Property
1	Rosenbrock	$f_1(x) = \sum_{i=1}^{N} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	−2.48	2.48	0	Unimodal
2	Griewangk	$f_2(x) = \sum_{i=1}^{N} \left(\frac{x_i^2}{4000}\right) - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	−50	50	0	Multimodal
3	Rastrigin	$f_3(x) = 10 \times N + \sum_{i=1}^{N} (x_i^2 - 10 \cdot \cos(2\pi x_i))$	−50	50	0	Multimodal

selected chaotic map for the *j*th dimension of the *i*th colony member. Fig. 5 depicts the pseudo-code of chaotic search for the bees.

3.3. Chaotic ABC 3 (CABC3)

CABC1 and CABC2 are combined, that is initial colony is generated by iterating the selected chaotic maps and chaotic search is applied in case of not obtaining improvements.

4. Test problems

Well-defined benchmark functions which are based on mathematical functions can be used as objective functions to measure and test the performance of optimization methods. The nature, complexity and other properties of these benchmark functions can be easily obtained from their definitions. The difficulty levels of most benchmark functions are adjustable by setting their parameters. From the standard set of benchmark problems available in the literature, three important functions one of which is unimodal (containing only one optimum) ant two of which are multimodal (containing many local optima, but only one global optimum) are considered to test the efficacy of the proposed methods. Table 2 shows the main properties of the selected benchmark functions used in the experiments.

5. Experimental results

Selected three benchmark problems are solved by simulating the ABC, CABC1, and CABC2 algorithms. Two criteria are applied to terminate the simulation of the algorithms: reaching maximum number of iterations which is set to a constant number and the second criterion is getting a minimum error.

All ABC was initialized in regions that include the global optimum for a fair evaluation. The algorithms were run for 100 times to catch their stochastic properties. In this experiment, maximum iteration number was set to 500 and the goal is not to find the global optimum values but to find out the potential of the algorithms. Algorithm success rate defined in Eq. (2) has been used for comparison of the results obtained from different ABC algorithms.

$$S = 100 \frac{NT_{successful}}{NT_{all}}\bigg|_{Q_{level}}. \tag{2}$$

Table 3
Success rates of ABC algorithms for test functions.

Q_{level}	Rosenbrock (*N* = 2)	Griewangk (*N* = 10)	Rastrigin (*N* = 10)
1.e−5	0	13	75
1.e−6	0	13	60

Table 4
Success rates of CABC algorithms using different chaotic maps for Rosenbrock (*N* = 2).

Q_{level}	CABC1	CABC2	CABC3
Logistic map			
1.e−5	0	6	6
1.e−6	0	4	4
Circle map			
1.e−5	1	5	4
1.e−6	1	4	4
Gauss map			
1.e−5	1	6	7
1.e−6	1	5	6
Henon map			
1.e−5	2	4	5
1.e−6	1	3	3
Sinusoidal iterator			
1.e−5	0	4	4
1.e−6	0	2	3
Sinus map			
1.e−5	0	6	5
1.e−6	0	5	5
Tent map			
1.e−5	0	6	6
1.e−6	0	4	5

Table 5
Success rates of CABC algorithms using different chaotic maps for Griewangk ($N = 10$).

Q_{level}	CABC1	CABC2	CABC3
Logistic map			
1.e−5	16	26	25
1.e−6	10	23	22
Circle map			
1.e−5	14	18	17
1.e−6	14	16	17
Gauss map			
1.e−5	18	26	23
1.e−6	8	23	21
Henon map			
1.e−5	18	28	28
1.e−6	13	21	26
Sinusoidal iterator			
1.e−5	19	25	23
1.e−6	14	18	20
Sinus map			
1.e−5	16	28	27
1.e−6	8	19	19
Tent map			
1.e−5	17	23	23
1.e−6	13	15	16

Table 6
Success rates of CABC algorithms using different chaotic maps for Rastrigin ($N = 10$).

Q_{level}	CABC1	CABC2	CABC3
Logistic map			
1.e−5	69	91	89
1.e−6	59	85	69
Circle map			
1.e−5	68	90	88
1.e−6	61	84	81
Gauss map			
1.e−5	76	95	91
1.e−6	58	84	82
Henon map			
1.e−5	65	89	89
1.e−6	46	82	86
Sinusoidal iterator			
1.e−5	72	88	89
1.e−6	70	79	86
Sinus map			
1.e−5	26	92	92
1.e−6	25	81	86
Tent map			
1.e−5	72	88	87
1.e−6	56	79	79

$N_{successful}$ is the number of trials, which found the solution on the Q_{level} in the allowable maximum iteration. N_{all} is the number of all trials. Q_{level} is the end condition to stop the algorithm, when it converges into Q_{level} tolerance.

The colony size for the algorithms has been selected as 20. Limit parameter for ABC is selected as 40. Table 3 depicts the success rates of ABC algorithms for the test functions. Success rates of CABC algorithms using different chaotic maps for Rosenbrock function are shown in Table 4. CABC algorithms have somewhat shown better performance than ABC algorithm for this test function. Especially, all of the results obtained from CABC2 and CABC3 algorithms are better than that from ABC algorithm.

Success rates of CABC algorithms using different chaotic maps for Griewangk and Rastrigin functions are shown in Tables 5 and Table 6, respectively. Similar to the results obtained from Rosenbrock test function, CABC algorithms have somewhat shown better performance than ABC algorithm. Especially, all of the results obtained from CABC2 and CABC3 algorithms are better than that from ABC algorithm.

6. Conclusions

In this paper different chaotic maps have been embedded to adapt the parameters of ABC algorithm. This has been done by using of chaotic number generators. Two new chaotic ABC algorithms have been proposed and seven chaotic maps have been analyzed in the benchmark functions. It has been detected that coupling emergent results in different areas, like those of ABC and complex dynamics, can improve the quality of results in some optimization problems and also that chaos may be a desired process as in real honey bee colony. It has been also shown that, these methods have somewhat increased the solution quality, that is in some cases they improved the global searching capability by escaping the local solutions. These proposed methods are new and more elaborated experiments may be performed with parallel or distributed implementation.

References

Abbass, H. A. (2001). MBO: Marriage in honey bees optimization – A Haplometrosis polygynous swarming approach. *IEEE Congress on Evolutionary Computation, 1*, 207–214.

Alatas, B., Akin, E., & Ozer, B. (2009). Chaos embedded particle swarm optimization algorithms. *Chaos, Solitons & Fractals*. doi:10.1016/j.chaos.2007.09.063.

Arena, P., Caponetto, R., Fortuna, L., Rizzo, A., & La Rosa, M. (2000). Self organization in non recurrent complex system. *International Journal of Bifurcation and Chaos, 10*(5), 1115–1125.

Baykasoglu, A., Ozbakir, L., & Tapkan, P. (2007). Artificial bee colony algorithm and its application to generalized assignment problem. In Felix T. S. Chan & Manoj Kumar Tiwari (Eds.), *Swarm intelligence: Focus on Ant and particle swarm optimization* (pp. 532). Itech Education and Publishing (chap. 8).

Coelho, L. S., & Mariani, V. C. (2008). Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. *Expert Systems with Applications, 34*, 1905–1913.

Han, F., Hu, J., Yu, X., & Wang, Y. (2007). Fingerprint images encryption via multi-scroll chaotic attractors. *Applied Mathematics and Computation, 185*(2), 931–939.

Heidari-Bateni, G., & McGillem, C. D. (1994). A Chaotic direct-sequence spread spectrum communication system. *IEEE Transaction on Communications, 42*(2/3/4), 1524–1527.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization technical report-TR06*. Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing, 8*, 687–697.

Manganaro, G., & de Gyvez, J. P. (1997). DNA computing based on chaos. In *IEEE International conference on evolutionary computation* (pp. 255–260). Piscataway, NJ: IEEE Press.

Pai, P. F., Yang, S. L., & Chang, P. T. (2009). Forecasting output of integrated circuit industry by support vector regression models with marriage honey-bees optimization algorithms. *Expert Systems with Applications, 36*(7), 10746–10751.

Pham, D. T., Kog, E., Ghanbarzadeh, A., Otri, S., Rahim, S., & Zaidi, M. (2006). The bees algorithm – A novel tool for complex optimisation problems. In *IPROMS 2006 proceeding 2nd international virtual conference on intelligent production machines and systems*. Oxford: Elsevier.

Schuster, H. G. (1988). *Deterministic chaos: An introduction* (2nd revised ed.). Federal Republic of Germany: Physick-Verlag, GmnH, Weinheim.

Suneel, M. (2006). Chaotic sequences for secure CDMA. *Ramanujan Institute for Advanced Study in Mathematics*, 1–4.

Wong, K., Man, K. P., Li, S., & Liao, X. (2005). More secure chaotic cryptographic scheme based on dynamic look-up table circuits. *Systems & Signal Processing, 24*(5), 571–584.

Yang, X. (2005). *Engineering optimizations via nature-inspired virtual bee algorithms, Lecture notes in computer science* (Vol. 3562, pp. 317–323). Springer-Verlag, GmbH.